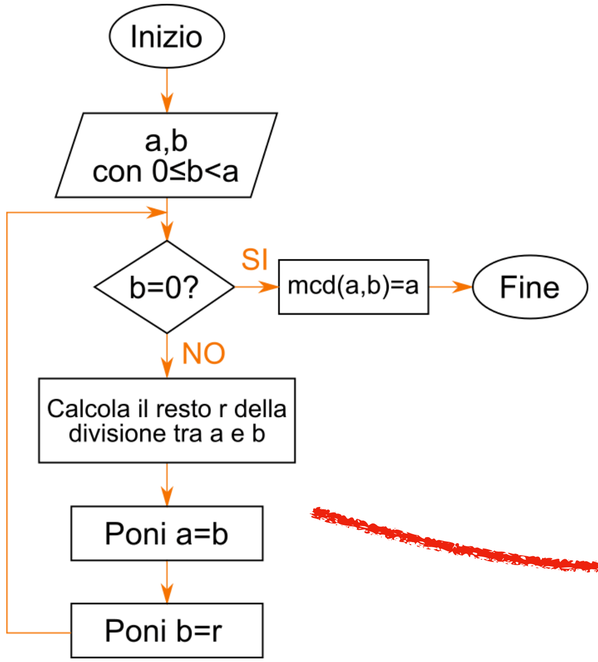
Lab 2

Dagli algoritmi ai programmi

ALGORITMO

Linguaggio alto livello (per Umani)



LINGUAGGIO di PROGRAMMAZIONE

- Consente la descrizione di algoritmi
- Usando costrutti comprensibili all'essere umano (q.m.)
- Traducibili automaticamente in linguaggio macchina

ESECUTORE

Linguaggio basso livello (per Macchine)

Linguaggi di Programmazione

Ada
Time-tested, safe and secure

python™

Kotlin

...about
700
programming languages
available!!!!

Java™

julia

Ruby

C++
Pros e Cons

Come fare a recensire una bicicletta se non si sa che cosa è un cambio?

Per noi un linguaggio varrebbe un altro ma:

- Lo userete in altri corsi (TNDS)
- È abbastanza a basso livello per consentire di vedere che cosa succede in memoria (capire che cosa sta facendo la macchina)
- È abbastanza a basso livello per non fare schifezze di nascosto (ogni riferimento a Python è totalmente voluto)
- (Può essere usato come linguaggio imperativo, evitando gli oggetti)

Strumenti di Lavoro

Ambiente: Linux, Windows, Mac, Android, non importa fintanto che potrete installare ed usare in modo comodo i seguenti strumenti:...

Editor di Testi: serve a scrivere il codice sorgente, ovvero un file di testo semplice (no immagini, formattazione strane, Emoji) contenente le istruzioni in linguaggio C++:

- Base (GEdit, VIM, EMACS) NO WORD!!!!
- Integrated Development Environment (IDE): Visual Studio Code, Eclipse, XCode (MAC)

Compilatore C++: un programma che prende in ingresso il file di testo prodotto con l'editor e lo traduce in linguaggio macchina, salvando la traduzione in un altro file. Ci sono diversi compilatori a disposizione:

- GNU Compiler Collection (gcc): contiene anche il compilatore C++
- clang (Apple)
- MinGW (porting (adattamento)) Windows del gcc
- Altro....

Workflow

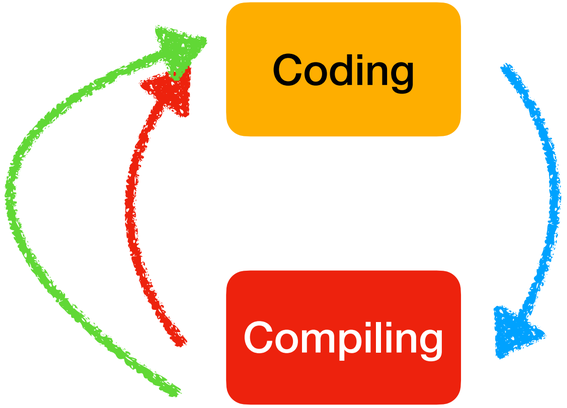
La tipica sessione di laboratorio consiste nello svolgimento di esercizi volti ad allenare la vostra capacità di :

- definire un algoritmo per la soluzione del problema
- verificare la correttezza dell'algoritmo
- determinare i costrutti del linguaggio necessari all'implementazione dell'algoritmo
- scrivere il programma che risolve il problema assegnato
- verificare la correttezza del codice scritto

La parte di scrittura del codice, in particolare, si può così schematizzare

Scritto codice nuovo, tutto ok,
passo al prossimo pezzo di soluzione

Scritto codice nuovo, errore sintassi/funzionamento,
provo a correggere



Dichiarazione variabili

Prepariamo i contenitori di informazione che serviranno durante l'esecuzione del programma per realizzare l'algoritmo (trasformazione Ingresso -> Uscita)

(alcuni) Tipi di dato (primitivi) del C++

- Int32 `-> int`
- Float32 `-> float`
- Float64 `-> double`
- character/byte `-> char`
- boolean `-> bool`
- Int64 `-> long int`

```
/*Qui comincia il programma ve
int main(){
    /*Dichiarazione variabili*/
    int a;
    int b;
    int c;


    /*Istruzioni*/
    a=5;
```

Istruzioni

Assegnamento, I/O

```
/*Istruzioni*/
a=5;
b=2*a;
c=a+b;

//Istruzione stampa a video
cout << endl << "risultato in c: " << c << endl;


/*fine programma*/
return 0;
```

Controllo Flusso calcolo

Sequenza...

Lettura e stampa dati

libreria <iostream>

- Per poter "dialogare" con un programma (fornire informazioni in ingresso/visualizzare risultati prodotti) è necessario usare due strumenti della libreria (leggi "collezione di strumenti") **<iostream>**.

- **`cin >> nomevariabile`**: legge da tastiera una sequenza di caratteri, li "interpreta" e registra quanto interpretato nella variabile **`nomevariabile`**

- Esempio:

```
int a;   //Dichiaro una variabile intera di nome a

cin >> a;   // Quando esegue questa istruzione il programma attende che l'utente scriva qualcosa
            // da tastiera e schiacci Invio.
            // Quando viene schiacciato Invio, il programma legge quanto scritto e lo registra (se compatibile) in a/
```

Stampa a video

- cout << "Messaggio: " << nomevariabile << endl;
- **`cout << "Il valore in a e`: " << a << endl;`**

Stampa a video la stringa **`"Il valore in a e`: 5"`** se in a è registrato il valore 5.

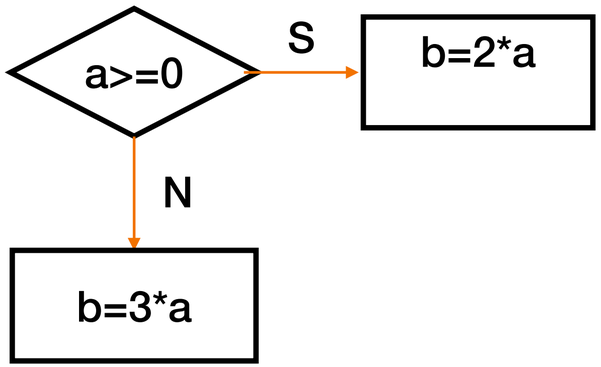
endl fa andare a capo (endline).

Se voglio comporre messaggi più articolati, posso sbizzarrirmi con la concatenazione di messaggi <<" ...." << e contenuti di variabili

"Dario e` alto "<< altezza <<" cm" << endl << "Peso (dichiarato): " << peso << " chili" << endl;

Strutture controllo: selezione

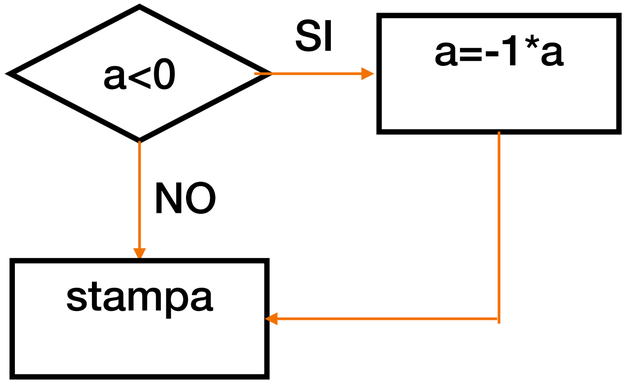
Selezione: if...else....



selezione.C

```
if( a>=0 ){ //Blocco 1
    b = 2*a;
}
else{ //Blocco 2
    b=3*a;
}
```

Selezione "monca": if...

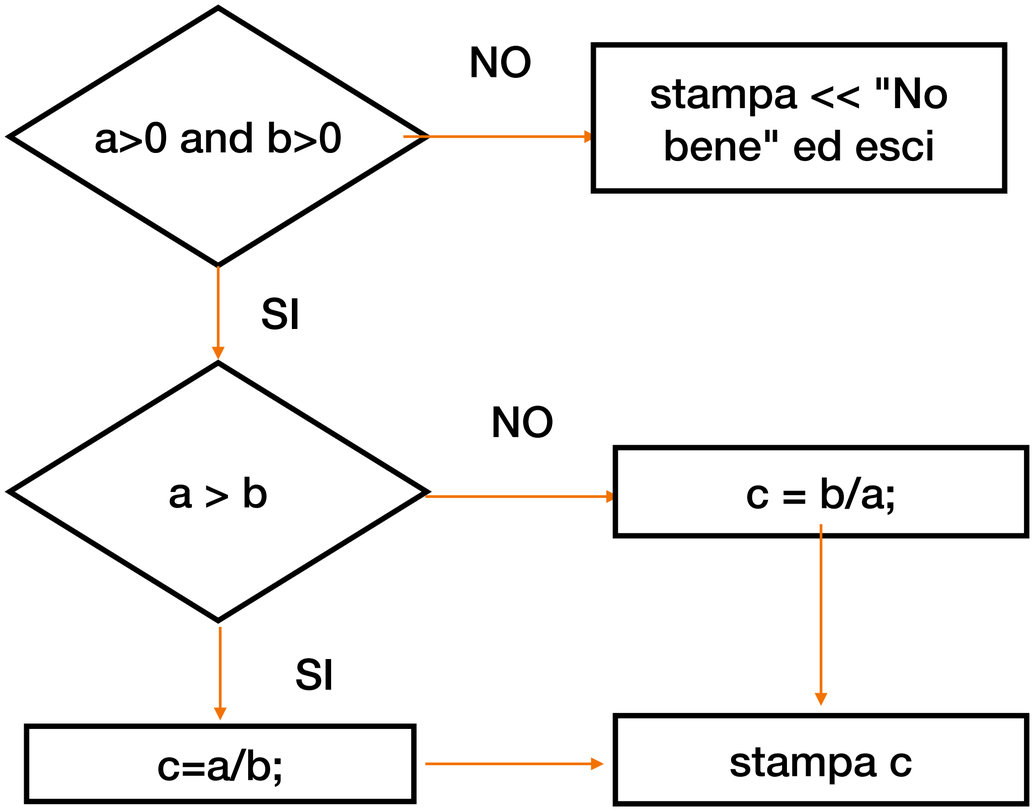


sel_monca.C

```
if( a < 0 ){ //Blocco 1
    a = -1*a;
}
 //No Blocco 2
 //Prosegue l'esecuzione

cout << endl << "Risultato, a= " << a << endl;
```

Selezione annidata



sel_ann.C

```
if (a>0 and b> 0){

    if (a>=b){
        c=a/b;
    }
    else{
        c=b/a;
    }
    cout << endl << "Risultato, c= " << c << endl;
}
else{
    cout << endl << "No bene! Esco";
    //L'istruzione return provoca l'uscita dalla funzione mai
    return -1; //Valore "restituito"....ci torniamo
}
```